DSC 140B - Homework 02

Due: Wednesday, April 19

Write your solutions to the following problems by either typing them up or handwriting them on another piece of paper. Unless otherwise noted by the problem's instructions, show your work or provide some justification for your answer. Homeworks are due via Gradescope at 11:59 PM.

Problem 1.

Suppose (just like in the last homework) that in a group of 1000 people, 600 currently live in California and 400 currently live in Texas. In any given year, 5% of the people living in California move to Texas, and 3% of the people living in Texas move to California. You may assume that the people do not move to any other states.

We can represent the current number of people living in California and Texas with a *population vector*:

$$\vec{p} = (\# \text{ in California}, \# \text{ in Texas})^T.$$

The initial situation described above is represented by the population vector $(600, 400)^T$.

a) Let $\vec{f}(\vec{p})$ be the linear transformation which takes in a current population vector, $\vec{p} = (c, t)^T$, and returns the population vector after one year has passed. In part (b) of the corresponding problem on the last homework, you should have found the following formula for $\vec{f}$ with respect to the standard basis:

$$\vec{f}(\vec{p}) = (.95c + .03t, \quad .05c + .97t)^T$$

Write the *matrix* A representing $\vec{f}$ with respect to the standard basis.

Solution: For the columns of A we use the vectors $\vec{f}(\hat{e}^{(1)})$ and $\vec{f}(\hat{e}^{(2)})$. So we start by computing them:

$$\begin{aligned}\vec{f}(\hat{e}^{(1)}) &= \vec{f}((1,0)^T) \\ &= (.95, .05)^T\end{aligned}$$

$$\begin{aligned}\vec{f}(\hat{e}^{(2)}) &= \vec{f}((0,1)^T) \\ &= (.03, .97)^T\end{aligned}$$

And so the matrix representing this transformation (in the standard basis) is:

$$A = \begin{pmatrix} .95 & .03 \\ .05 & .97 \end{pmatrix}$$

b) Using a matrix multiplication, find the population vector after one year has passed, given that the initial population vector is $(600, 400)^T$. Your result should not contain decimals.

Solution:

$$\begin{aligned} A\vec{x} &= \begin{pmatrix} .95 & .03 \\ .05 & .97 \end{pmatrix} (600, 400) \\ &= \begin{pmatrix} .95 \times 600 + .03 \times 400 \\ .05 \times 600 + .97 \times 400 \end{pmatrix} \\ &= \begin{pmatrix} 582 \\ 418 \end{pmatrix} \end{aligned}$$

c) In part (f) of the last homework, we saw that two eigenvectors of A are

$$\vec{u}^{(1)} = \begin{pmatrix} 375 \\ 625 \end{pmatrix} \qquad \vec{u}^{(2)} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

Verify that these are eigenvectors of the matrix A by performing the matrix multiplication.

Solution:

$$\begin{aligned} A\vec{u}^{(1)} &= \begin{pmatrix} .95 & .03 \\ .05 & .97 \end{pmatrix} \begin{pmatrix} 375 \\ 625 \end{pmatrix} \\ &= \begin{pmatrix} .95 \times 375 + .03 \times 625 \\ .05 \times 375 + .97 \times 625 \end{pmatrix} \\ &= \begin{pmatrix} 356.25 + 18.75 \\ 18.75 + 606.25 \end{pmatrix} \\ &= \begin{pmatrix} 356.25 + 18.75 \\ 18.75 + 606.25 \end{pmatrix} \\ &= \begin{pmatrix} 375 \\ 625 \end{pmatrix} \end{aligned}$$

So $\vec{u}^{(1)}$ is an eigenvector with eigenvalue 1.

$$\begin{aligned} A\vec{u}^{(2)} &= \begin{pmatrix} .95 & .03 \\ .05 & .97 \end{pmatrix} \begin{pmatrix} 1 \\ -1 \end{pmatrix} \\ &= \begin{pmatrix} .95 \times 1 + .03 \times -1 \\ .05 \times 1 + .97 \times -1 \end{pmatrix} \\ &= \begin{pmatrix} .95 - .03 \\ .05 - .97 \end{pmatrix} \\ &= \begin{pmatrix} .92 \\ -.92 \end{pmatrix} \\ &= 0.92 \begin{pmatrix} 1 \\ -1 \end{pmatrix} \end{aligned}$$

So $\vec{u}^{(2)}$ is an eigenvector with eigenvalue 0.92.

d) Write the matrix $A_{\mathcal{U}}$ of the linear transformation $\vec{f}$ with respect to the basis $\mathcal{U} = \{\vec{u}^{(1)}, \vec{u}^{(2)}\}$.

Solution: The matrix has as its columns $[f(\vec{u}^{(1)})]_{\mathcal{U}}$ and $[f(\vec{u}^{(2)})]_{\mathcal{U}}$.

We know already that $f(\vec{u}^{(1)}) = \vec{u}^{(1)}$, which means

$$[f(\vec{u}^{(1)})]_{\mathcal{U}} = (1,0)^T.$$

Similarly, $f(\vec{u}^{(2)}) = 0.92\vec{u}^{(2)}$, so

$$[f(\vec{u}^{(2)})]_{\mathcal{U}} = (0,.92)^T.$$

Therefore:

$$A_{\mathcal{U}} = \begin{pmatrix} 1 & 0 \\ 0 & 0.92 \end{pmatrix}$$

e) In part (f) of the last homework, you found that the initial population vector $\vec{x} = (600, 400)^T$ expressed in the new basis has coordinates $[\vec{x}]_{\mathcal{U}} = (1, 225)^T$.

Compute $A_{\mathcal{U}}[\vec{x}]_{\mathcal{U}}$ and then convert the resulting to a coordinate vector in the standard basis.

Hint: your result should be familiar.

Solution:

$$\begin{aligned} A_{\mathcal{U}}[\vec{x}]_{\mathcal{U}} &= \begin{pmatrix} 1 & 0 \\ 0 & .92 \end{pmatrix} \begin{pmatrix} 1 \\ 225 \end{pmatrix} \\ &= \begin{pmatrix} 1 \\ .92 \times 225 \end{pmatrix} \\ &= \begin{pmatrix} 1 \\ 207 \end{pmatrix} \end{aligned}$$

This is telling us that $\vec{f}(\vec{x}) = 1\vec{u}^{(1)} + 207\vec{u}^{(2)}$. Therefore, in the standard basis:

$$\begin{aligned} \vec{f}(\vec{x}) &= \vec{u}^{(1)} + 207\vec{u}^{(2)} \\ &= \begin{pmatrix} 375 \\ 625 \end{pmatrix} + 207 \begin{pmatrix} 1 \\ -1 \end{pmatrix} \\ &= \begin{pmatrix} 375 + 207 \\ 625 - 207 \end{pmatrix} \\ &= \begin{pmatrix} 582 \\ 418 \end{pmatrix} \end{aligned}$$

Problem 2.

Let $g(\vec{x}) = g(x_1, x_2) = 4x_1^2 + 3x_2^2 + 10x_1x_2$, where we've defined $\vec{x} = (x_1, x_2)^T$. In this problem, we will consider maximizing g subject to the constraint $x_1^2 + x_2^2 = 1$.

You saw how to solve optimization problems like this in your multivariate calculus class using the method of *Lagrange multipliers*. Informally-speaking, the idea behind Lagrange multipliers is that the gradient vector of g and the gradient of the constraint $x_1^2 + x_2^2 - 1$ should be parallel at a constrained optimum. Since two vectors $\vec{a}$ and $\vec{b}$ are parallel if and only if $\vec{a} = \lambda\vec{b}$ for some λ, and since the gradient of the constraint is simply $(2x_1, 2x_2)^T = 2\vec{x}$, this means that a local optimum should satisfy $\nabla g(\vec{x}) = 2\lambda\vec{x}$. This looks similar to the eigenvector equation $A\vec{x} = \lambda\vec{x}$; in this problem we'll make the connection clearer.

a) The Lagrange multiplier approach says that we should define the *Lagrangian*:

$$\mathcal{L}(x_1, x_2, \lambda) = g(\vec{x}) - \lambda(x_1^2 + x_2^2 - 1)$$

We then solve the system of three equations in three unknowns:

$$\frac{\partial \mathcal{L}}{\partial x_1} = 0$$

$$\frac{\partial \mathcal{L}}{\partial x_2} = 0$$

$$\frac{\partial \mathcal{L}}{\partial \lambda} = 0$$

Write out and solve this system for x_1, x_2, and λ.

Hint: Try to get a formula for x_1^2 in terms of λ only, and same for x_2^2. When you get to this point, you will be able to substitute your formulas for x_1^2 and x_2^2 into $\partial\mathcal{L}/\partial\lambda = 0$ to get a function of the form

$$\frac{a_1}{(b_1\lambda + c_1)^2 + d_1} + \frac{a_2}{(b_2\lambda + c_2)^2 + d_2} - 1 = 0,$$

where the a, b, c, d's are all constants. We want to solve this for λ, which is not easy to do analytically. Instead, solve it numerically using `scipy.optimize.fsolve`, or similar. Once you've solved for λ, you can plug it back in to your equations for x_1^2 and x_2^2.

Solution: The partial derivatives are:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial x_1} &= 8x_1 + 10x_2 - 2\lambda x_1 \\
\frac{\partial \mathcal{L}}{\partial x_2} &= 6x_2 + 10x_1 - 2\lambda x_2 \\
\frac{\partial \mathcal{L}}{\partial \lambda} &= x_1^2 + x_2^2 - 1
\end{aligned}$$

Setting the last equations to zero and solving for (alternately) x_1 and x_2 yields:

$$\begin{aligned}
x_1 &= \sqrt{1 - x_2^2} \\
x_2 &= \sqrt{1 - x_1^2}
\end{aligned}$$

We can plug these into the first two equations to reduce the number of variables in each. For instance, the first equation becomes:

$$\begin{aligned}
&8x_1 + 10x_2 - 2\lambda x_1 = 0 \\
&\implies (8 - 2\lambda)x_1 + 10\sqrt{1 - x_1^2} = 0 \\
&\implies 10\sqrt{1 - x_1^2} = (2\lambda - 8)x_1
\end{aligned}$$

Squaring both sides:

$$\begin{aligned}
&\implies 100(1 - x_1^2) = (2\lambda - 8)^2 x_1^2 \\
&\implies 100 - 100x_1^2 = (2\lambda - 8)^2 x_1^2 \\
&\implies 100 = \left[(2\lambda - 8)^2 + 100\right] x_1^2 \\
&\implies x_1^2 = 100/\left[(2\lambda - 8)^2 + 100\right]
\end{aligned}$$

Similarly for x_2, we have:

$$\begin{aligned}
&6x_2 + 10x_1 - 2\lambda x_2 = 0 \\
&\implies (6 - 2\lambda)x_2 + 10\sqrt{1 - x_2^2} = 0 \\
&\implies 10\sqrt{1 - x_2^2} = (2\lambda - 6)x_2
\end{aligned}$$

Squaring both sides:

$$\begin{aligned}
&\implies 100(1 - x_2^2) = (2\lambda - 6)^2 x_2^2 \\
&\implies 100 - 100x_2^2 = (2\lambda - 6)^2 x_2^2 \\
&\implies 100 = \left[(2\lambda - 6)^2 + 100\right] x_2^2 \\
&\implies x_2^2 = 100/\left[(2\lambda - 6)^2 + 100\right]
\end{aligned}$$

Plugging these back into the formula for $\partial\mathcal{L}/\partial\lambda = 0$, we get:

$$\begin{aligned}
&\partial\mathcal{L}/\partial\lambda = 0 \\
&\implies x_1^2 + x_2^2 - 1 = 0 \\
&\implies \frac{100}{(2\lambda - 8)^2 + 100} + \frac{100}{(2\lambda - 6)^2 + 100} - 1 = 0
\end{aligned}$$

Solving this by hand is not possible. Instead, we can solve it numerically using a solver, such as `scipy.optimize.fsolve`. We also plot the function so that we can get a sense for where its roots are:

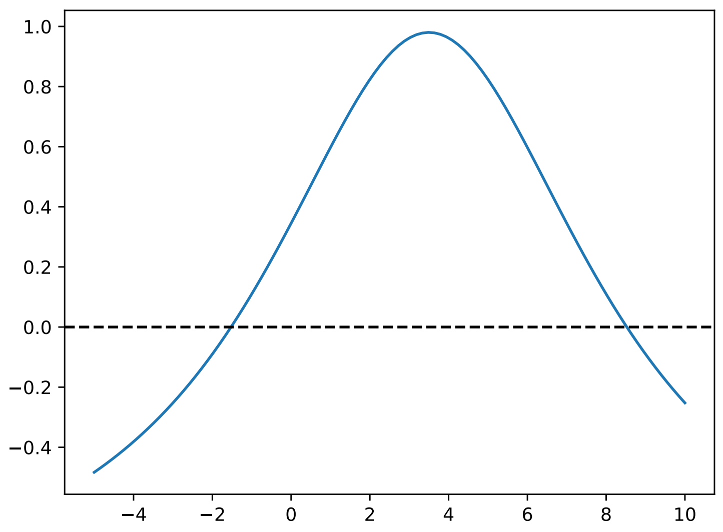
```python
import numpy as np
import matplotlib.pyplot as plt
import scipy.optimize

def f(x):
    return (
            100 / ((2*x - 8)**2 + 100)
            +
            100 / ((2*x - 6)**2 + 100)
            -
            1
        )

x = np.linspace(-5, 10, 100)
plt.plot(x, f(x))
plt.axhline(0, color='black', linestyle='--')
plt.show()

print(scipy.optimize.fsolve(f, -2))
print(scipy.optimize.fsolve(f, 9))
```

Now, there are multiple roots of the function, as the plot shows:



We guess that they are around -2 and 8. Using these as the starting location in `scipy.optimize.fsolve`, we get two approximated roots: -1.525 and 8.525.

We plug these back into our formulas for x_1^2 and x_2^2:

$$x_1^2 = 100/\left[(2\lambda - 8)^2 + 100\right] =$$
$$x_2^2 = 100/\left[(2\lambda - 6)^2 + 100\right] =$$

With $\lambda = -1.525$, we get:

$$x_1 = \pm 0.671$$
$$x_2 = \pm 0.741$$

With $\lambda = 8.525$, we get:

$$x_1 = \pm 0.741$$
$$x_2 = \pm 0.671$$

This seems to yield 8 possible combinations of x_1, x_2, λ, but not all combinations actually solve the original system of equations. For example, $x_1 = .67$, $x_2 = .74$, and $\lambda = -1.525$ does not satisfy $\partial L/\partial x_1 = 0$. Filtering out the four non-solutions, we have:

x_1	x_2	λ	$g(x_1, x_2)$
0.67	-0.74	-1.525	-1.52
-0.67	0.74	-1.525	-1.52
0.74	0.67	8.525	8.52
-0.74	-0.67	8.525	8.52

b) The equation $g(\vec{x}) = 4x_1^2 + 3x_2^2 + 10x_1x_2$ can be written in matrix-vector form as $g(\vec{x}) = \vec{x}^T A\vec{x}$ for an appropriately-defined matrix, A. Find this matrix A, and show that the matrix form is equivalent to the original form.

You can assume that A is symmetric.

Solution: Define

$$A = \begin{pmatrix} 4 & 5 \\ 5 & 3 \end{pmatrix}$$

We have:

$$\begin{aligned} A\vec{x} &= \begin{pmatrix} 4 & 5 \\ 5 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \\ &= \begin{pmatrix} 4x_1 + 5x_2 \\ 5x_1 + 3x_2 \end{pmatrix} \end{aligned}$$

So:

$$\begin{aligned} \vec{x}^T A\vec{x} &= \begin{pmatrix} x_1 & x_2 \end{pmatrix} \begin{pmatrix} 4x_1 + 5x_2 \\ 5x_1 + 3x_2 \end{pmatrix} \\ &= 4x_1^2 + 5x_1x_2 + 5x_1x_2 + 3x_2^2 \\ &= 4x_1^2 + 10x_1x_2 + 3x_2^2 \end{aligned}$$

c) Using whatever method you choose (e.g., numpy), compute the eigenvectors and eigenvalues of A. Show that the eigenvectors are the same as your solution to part (a).

Solution:

```
>>> import numpy as np
>>> A = np.array([[4, 5], [5, 3]])
>>> evals, evecs = np.linalg.eigh(A)
>>> print(evals)
[-1.52493781  8.52493781]
>>> print(evecs)
[[ 0.67100532 -0.74145253]
 [-0.74145253 -0.67100532]]
```

The columns of the `evecs` array are the two eigenvectors of A. We have two solutions: $\hat{u}^{(1)} = (0.671, -0.741)^T$ and $\hat{u}^{(2)} = (-0.741, -0.671)^T$. These are the same as the solutions to the first part. Note that in the first part, we also have $(-0.671, 0.741)^T$ and $(0.741, 0.671)^T$, but these are just $-\hat{u}^{(1)}$ and $-\hat{u}^{(2)}$, and are therefore not really "different" solutions.

d) We saw in lecture that a matrix can be interpreted as the representation of a linear transformation $\vec{f}(\vec{x})$. It turns out that A represents the *gradient* of $\vec{g}$.

Show that A represents the linear transformation $\vec{f}(\vec{x}) = \frac{1}{2}\nabla g(\vec{x})$, $\nabla g(\vec{x}) = (\partial g/\partial x_1,$ where $\partial g/\partial x_2)^T$ is the *gradient* of g.

Solution: We have:

$$\frac{\partial g}{\partial x_1} = 8x_1 + 6x_2 \qquad \frac{\partial g}{\partial x_2} = 6x_2 + 6x_1$$

So the gradient vector is:

$$\nabla g(\vec{x}) = \begin{pmatrix} 8x_1 + 10x_2 \\ 10x_2 + 6x_1 \end{pmatrix}$$

and

$$\vec{f}(\vec{x}) = \frac{1}{2}\nabla g(\vec{x}) = \begin{pmatrix} 4x_1 + 5x_2 \\ 5x_2 + 3x_1 \end{pmatrix}$$

Whereas:

$$A\vec{x} = \begin{pmatrix} 4 & 5 \\ 5 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 4x_1 + 5x_2 \\ 5x_1 + 3x_2 \end{pmatrix} = \frac{1}{2}\nabla g(\vec{x})$$

Therefore, for a function g of the form $ax_1^2 + bx_2^2 + cx_1x_2$, the gradient is a linear transformation that can be computed by a matrix multiplication, and the method of Lagrange multipliers is equivalent to finding an eigenvector of the matrix representing the gradient.

Problem 3.

The power method is a simple approach to computing an eigenvector of a matrix A. The method is as follows:

1. Initialize $\vec{x}^{(0)}$ arbitrarily (e.g., randomly).
2. Repeat until convergence:
   (a) Set $\vec{x}^{(i+1)} = (A\vec{x}^{(i)})/\|A\vec{x}^{(i)}\|$.

Eventually, $\vec{x}^{(i)}$ will be the top eigenvector of A.

In this problem, we'll develop an intuition for why this works.

a) The algorithm described above normalizes at each step for numerical precision. This complicates the analysis slightly. Instead, suppose for now that at every step we do not normalize; that is $\vec{x}^{(i+1)} = A\vec{x}^{(i)}$. In other words, after k iterations, we'll have $\vec{x}^{(k)} = A^k\vec{x}^{(0)}$. Only at the very end do we normalize by dividing by $A^k\vec{x}^{(0)}$. That is, after k iterations we'll return $A^k\vec{x}^{(0)}/\|A^k\vec{x}^{(0)}\|$.

Suppose that A is a $d \times d$ symmetric matrix. Let the eigendecomposition of $\vec{x}^{(0)}$ in terms of A be $\vec{x}^{(0)} = a_1\hat{u}^{(1)} + \ldots + a_d\hat{u}^{(d)}$, where $\{\hat{u}^{(1)}, \ldots, \hat{u}^{(d)}\}$ form an orthonormal basis of eigenvectors of A. You may assume that $A\hat{u}^{(i)} = \lambda_i\hat{u}^{(i)}$, where λ_i is the eigenvalue associated with $\hat{u}^{(i)}$, that $|\lambda_1| > |\lambda_2| > |\lambda_3| \cdots$, etc., and that $a_1 \neq 0$. For simplicity, you may assume that all of the eigenvalues are positive, although this is actually not necessary for the proof.

Find a formula for $\|A^k\vec{x}^{(0)}\|$ that involves only $a_1, \ldots, a_d$ and the eigenvalues $\lambda_1, \ldots, \lambda_d$.

Solution:

$$\begin{aligned} A^k\vec{x}^{(0)} &= A^k(a_1\hat{u}^{(1)} + \ldots + a_d\hat{u}^{(d)}) \\ &= a_1A^k a_1\hat{u}^{(1)} + \ldots + a_dA^k\hat{u}^{(d)}) \end{aligned}$$

If $\hat{u}^{(i)}$ is an eigenvector of A, then $A\hat{u}^{(i)} = \lambda_i\hat{u}^{(i)}$, by definition. It follows that $A^k\hat{u}^{(i)} = \lambda_i^k\hat{u}^{(i)}$:

$$\begin{aligned} &= a_1A^k a_1\hat{u}^{(1)} + \ldots + a_dA^k\hat{u}^{(d)} \\ &= a_1\lambda_1^k\hat{u}^{(1)} + \ldots + a_d\lambda_d^k\hat{u}^{(d)} \end{aligned}$$

Now, for any vector $\vec{v}$, $\|\vec{v}\| = \sqrt{\vec{v}\cdot\vec{v}}$. So $\|A^k\vec{x}^{(0)}\|^2 = (A^k\vec{x}^{(0)}) \cdot (A^k\vec{x}^{(0)})$. We have:

$$\|A^k\vec{x}^{(0)}\|^2 = (a_1\lambda_1^k\hat{u}^{(1)} + \ldots + a_d\lambda_d^k\hat{u}^{(d)}) \cdot (a_1\lambda_1^k\hat{u}^{(1)} + \ldots + a_d\lambda_d^k\hat{u}^{(d)})$$

Any term of the form $\hat{u}^{(i)} \cdot \hat{u}^{(j)}$ with will be zero when $i \neq j$ because $\hat{u}^{(i)}$ and $\hat{u}^{(j)}$ are orthogonal, or one when $i = j$. And so we are left with only the terms where $i = j$. That is:

$$= a_1^2\lambda_1^{2k} + \ldots + a_d^2\lambda_d^{2k}$$

Therefore $\|A^k\vec{x}^{(0)}\| = \sqrt{a_1^2\lambda_1^{2k} + \ldots + a_d^2\lambda_d^{2k}}$

b) Show that

$$\lim_{k\to\infty} \frac{A^k\vec{x}^{(0)}}{\|A^k\vec{x}^{(0)}\|} = \hat{u}^{(1)}.$$

That is, the result is the top eigenvector of A (the eigenvector whose eigenvalue is largest in absolute value).

Note: a proof by calculation is needed to earn full credit, but an informal argument with some calculation will earn almost all of the credit.

Solution: We have:

$$\begin{aligned}
\lim_{k\to\infty} \frac{A^k\vec{x}^{(0)}}{\|A^k\vec{x}^{(0)}\|} &= \lim_{k\to\infty} \frac{a_1\lambda_1^k\hat{u}^{(1)} + \ldots + a_d\lambda_d^k\hat{u}^{(d)}}{\sqrt{a_1^2\lambda_1^{2k} + \ldots + a_d^2\lambda_d^{2k}}} \\
&= \lim_{k\to\infty} \left(\frac{a_1\lambda_1^k\hat{u}^{(1)}}{\sqrt{a_1^2\lambda_1^{2k} + \ldots + a_d^2\lambda_d^{2k}}} + \ldots + \frac{a_d\lambda_1^k\hat{u}^{(d)}}{\sqrt{a_1^2\lambda_1^{2k} + \ldots + a_d^2\lambda_d^{2k}}} \right) \\
&= \lim_{k\to\infty} \frac{a_1\lambda_1^k\hat{u}^{(1)}}{\sqrt{a_1^2\lambda_1^{2k} + \ldots + a_d^2\lambda_d^{2k}}} + \ldots + \lim_{k\to\infty} \frac{a_d\lambda_d^k\hat{u}^{(d)}}{\sqrt{a_1^2\lambda_1^{2k} + \ldots + a_d^2\lambda_d^{2k}}}
\end{aligned}$$

Consider the jth term, which looks like:

$$\lim_{k\to\infty} \frac{a_j\lambda_j^k\hat{u}^{(j)}}{\sqrt{a_1^2\lambda_1^{2k} + a_2^2\lambda_2^{2k} \ldots + a_d^2\lambda_d^{2k}}}$$

We can show that this is zero if $j \neq 1$ and $\hat{u}^{(1)}$ if $j = 1$. The intution is that, when k is large, the denominator will look like $|a_1\lambda_1^k|$, since $|\lambda_1|$ is the largest out of all the eigenvalues. Then the limit will look like $|\lambda_j/\lambda_1|^k$, and since $|\lambda_j/\lambda_1| < 1$, this will be zero.

Let's try it. Factoring out an $a_1^2\lambda_1^{(2k)}$ from every term in the denominator:

$$\begin{aligned}
\lim_{k\to\infty} \frac{a_j\lambda_j^k\hat{u}^{(j)}}{\sqrt{a_1^2\lambda_1^{2k} + a_2^2\lambda_2^{2k}\ldots + a_d^2\lambda_d^{2k}}} &= \lim_{k\to\infty} \frac{a_j\lambda_j^k\hat{u}^{(j)}}{\sqrt{a_1^2\lambda_1^{2k}\left(1 + \frac{a_2^2\lambda_2^{2k}}{a_1^2\lambda_1^{2k}} + \ldots + \frac{a_d^2\lambda_d^{2k}}{a_1^2\lambda_1^{2k}}\right)}} \\
&= \lim_{k\to\infty} \frac{a_j\lambda_j^k\hat{u}^{(j)}}{|a_1\lambda_1^k|\sqrt{1 + \frac{a_2^2\lambda_2^{2k}}{a_1^2\lambda_1^{2k}} + \ldots + \frac{a_d^2\lambda_d^{2k}}{a_1^2\lambda_1^{2k}}}} \\
&= \frac{\lim_{k\to\infty}(a_j\lambda_j^k\hat{u}^{(j)})/|a_1\lambda_1^k|}{\lim_{k\to\infty}\sqrt{1 + \frac{a_2^2\lambda_2^{2k}}{a_1^2\lambda_1^{2k}} + \ldots + \frac{a_d^2\lambda_d^{2k}}{a_1^2\lambda_1^{2k}}}}
\end{aligned}$$

The limit in the denominator will be one, so we're left with:

$$\begin{aligned}
&= \lim_{k\to\infty} (a_j\lambda_j^k\hat{u}^{(j)})/|a_1\lambda_1^k| \\
&= \lim_{k\to\infty} \frac{a_j\lambda_j^k\hat{u}^{(j)}}{|a_1\lambda_1^k|}
\end{aligned}$$

If $j = 1$, then $a_j\lambda_j^k/|a_1\lambda_1| = 1$, and the limit will simply be $\hat{u}^{(1)}$. But if $j \neq 1$, then $|\lambda_j/\lambda_1| < 1$, and so the limit of $|\lambda_j/\lambda_1|^k$ will be zero.

c) Let A be the matrix

$$\begin{pmatrix} 3 & 7 & 2 \\ 7 & 9 & -3 \\ 2 & -3 & 1 \end{pmatrix}.$$

Implement the power method (the original version, which normalizes at each step) in code, and use it to compute the top eigenvector of A. Attach your code here.

Solution:

```python
import numpy as np

A = np.array([
    [3, 7, 2],
    [7, 9, -3],
    [2, -3, 1]
])

def power_method(A, x0, threshold=1e-5):
    eps = float('inf')
    x0 = x0 / np.linalg.norm(x0)
    while eps > threshold:
        x = A @ x0
        x = x / np.linalg.norm(x)
        eps = np.linalg.norm(x - x0)
        x0 = x

    return x0
```

```
evals, evecs = np.linalg.eigh(A)

print(power_method(A, np.array([1, 1, 1])))
print(evecs)
```

The eigenvector it outputs is $(0.52582284, 0.84271066, -0.11553823)^T$.

d) Using another method (e.g., `numpy.linalg.eigh`), compute the top eigenvector of A and verify that your implementation of the power method agrees.

Solution: Using `np.linalg.eigh`, we get the following eigenvectors: $\hat{u}^{(1)} = (0.66, -.49, -.57)^T$, $\hat{u}^{(2)} = (0.53, -.22, -.84)^T$, $\hat{u}^{(3)} = (-0.56, .81, .q11)^T$,

The power method found the third of these, which is the one with the largest eigenvalue (in absolute value).